UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/729,767	12/08/2003	Richard P. Himmer	I20 04983US	3406

128 7590 09/30/2008

HONEYWELL INTERNATIONAL INC.
101 COLUMBIA ROAD
P O BOX 2245
MORRISTOWN, NJ 07962-2245

EXAMINER	
WANG, BEN C	

ART UNIT	PAPER NUMBER
2192	

MAIL DATE	DELIVERY MODE
09/30/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/729,767	HIMMER ET AL.	
	Examiner	Art Unit	
	BEN C. WANG	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *11 June 2008*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-8 and 16-17* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-8 and 16-17* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Applicant's amendment dated June 11, 2008, responding to the Office action mailed March 13, 2008 provided in the rejection of claims 1-8 and 16-17, wherein claims 1-2, 5, 7, and 16-17 have been amended.

Claims 1-8 and 16-17 remain pending in the application and which have been fully considered by the examiner.

The status of claim 8 was inadvertently listed as "Original". It should be corrected as "Previously Presented".

Applicant's arguments with respect to claims currently amended have been fully considered but are moot in view of the new grounds of rejection – see *Al-Khudair et al.* - art made of record, as applied hereto.

2. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however,

will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Claim Rejections – 35 USC § 103(a)

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1-8 and 17 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hammack et al. (Pat. No. US 6,449,624 B1) in view of Al-Khudair et al., "Object-Oriented VErsioning in a Concurrent Engineering Design Environment", 2001, Springer-Verlag Berlin Heidelberg, pp. 105-125 (hereinafter 'Al-Khudair' - art made of record)

4. **As to claim 1** (Currently Amended), Hammack discloses a source control system for a process control system (e.g., Fig. 1 – a process control system; Col. 3, Lines 49-52 – a process control system includes a process controller connected to one or more host workstations or computers via a communication network such as an Ethernet connection or the like), comprising:

- a processor in a process control system (e.g., Col. 2, Lines 36-39 – the inventive system further includes a configuration routine and a version control routine, both

of which are stored in the computer-readable medium and configured to be executed by the processor);

- a database accessible by said processor to store information associated with an object under source control to be checked-out (e.g., Col. 12, Lines 66 through Col. 13, Lines 7 – the VCAT [Version Control and Audit Trail system] system may further include functionality that permits the user to request a status update for all of the items in the configuration database to ensure that those items that are checked out are indicated as such via checkmark or the like); and
- a check-out function operable on said processor to check-out said object (e.g., Fig. 6 – element of 120 – "Check Out" , "CheckOut Recursive"); to use said information to determine whether any dependent objects exist, and to automatically check-out said existing dependent objects (e.g., Col. 11, Lines 24-29 – the VCAT system preferably determines during each check-out operation which other versionable items need to be checked out in order to modify the configuration of an item; the modification of these other versionable items may be referred to as "consequential changes.", 45-48 – because the configuration of the process is set forth in a hierarchal manner, the VCAT system must allow for checking out items having subordinate items that are also versionable).

Further, Hammack discloses the data stored in an XML document is accessed in accordance with an object model that provides for parsing the document to create a data tree structure having a plurality of nodes associated with the version control data (e.g., Col. 22, Lines 52-58), but does not explicitly disclose wherein said object is a user

defined template that is derived from a preconfigured object, and wherein said existing dependent objects are children user defined templates of said object being checked out or instances of said object being checked out or of said children user defined templates.

However, in an analogous art of *Object-Orient Versioning in a Concurrent Engineering Design Environment*, Al-Khudair discloses wherein said object is a user defined template that is derived from a preconfigured object, and wherein said existing dependent objects are children user defined templates of said object being checked out or instances of said object being checked out or of said children user defined templates (e.g., P. 107, Lines 2-3 – The checkin and checkout operations are used to deposit and retrieve design objects to/from workspaces; Fig. 7 – An object-oriented model of configuration versions; Sec. 4 – An Object-Oriented Model for Configuration Versioning, 4.1 The Model - … an object-oriented model which supports *the semantic extension of versions to cope with configuration versions* as shown in Fig. 7. This model is independent of the content of the design object and can represent a totally versioned configuration (i.e. both the configuration and its components are versioned). This enables the version model to be sufficiently generic so that it can be applied to a variety of engineering domains …; P. 113, 1st full-Para – *Configuration versioning* is the creation of a new version of the configuration … the configuration state before the change is retained as well as the new state (i.e., derived from), allowing multiple versions of a configuration to co-exist. The evolution of a configuration is represented as a hierarchy called a *configuration-derivation graph* (CDG). A *generic configuration* (i.e., preconfigured object) represents an abstract configuration which is associated with

the versioned configuration … Configuration versioning allows the instance of each class (i.e., derived from) in the standard CCH (Composite Class Hierarchy) to have multiple versions (i.e., dependent objects) …)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Al-Khudair into the Hammack's system to further provide wherein said object is a user defined template that is derived from a preconfigured object, and wherein said existing dependent objects are children user defined templates of said object being checked out or instances of said object being checked out or of said children user defined templates in the Hammack system.

The motivation is that it would further enhance the Hammack's system by taking, advancing and/or incorporating Al-Khudair's system which offers significant advantages to determine if a dynamically bound configuration version is consistent with its design goals and its assembly has no design conflicts between its components' versions as once suggested by Al-Khudair (e.g., Abstract)

5. **As to claim 2** (Currently Amended) (incorporating the rejection in claim 1), Hammack discloses the system further comprising: a propagation function operable on said processor to propagate changes made to said object being checked out to said existing dependent objects, when said object is saved (e.g., Col. 29, Lines 11-16 – the configuration management system wherein the configuration routine is adapted to be executed by the processor to make changes to a first process control element and to

propagate changes to other process control elements that are affected by the changes made to the first process control element)

6. **As to claim 3** (Original) (incorporating the rejection in claim 1), Hammack discloses the system wherein said stored information includes a reference to a parent object (e.g., TABLE 4 – Recover/Purge Dialog Window, 1st entry – items – list all subordinate items deleted based upon a selected parent; Col. 23, Lines 36-41, 49-56)

7. **As to claim 4** (Original) (incorporating the rejection in claim 1), Hammack discloses the system wherein said stored information is at least one selected from the group consisting of: a name, a version number, a type and a status (e.g., Col. 20, Lines 15-20 – Furthermore, it is preferred that key words and labels be utilized to identify attributes such as object type and properties; examples of property labels are "NAME" and "DESCRIPTION"; Col. 13, Lines 58-66 – data representative of each prior configuration of an item is stored in the version control database together with data reflective of a version assigned thereto; the version is preferably identified by number, but may be indicated in any other manner)

8. **As to claim 5** (Currently Amended), Hammack discloses a method of automatic check-out for a source control system in a process control system, comprising:

- storing information associated with an object (e.g., Fig. 3; Col. 6, Lines 25-43 – with reference now to Fig. 3, the data stored in the configuration database may

be presented to a user via a configuration database administrative interface such as Delta V® Explorer, which will hereinafter be referred to as "the Explorer system"; the Explorer system sets forth a configuration hierarchy in a windows-type environment having a suite of configuration tools for modifying the elements of the hierarchy);

- receiving a request from a user to check-out said object (e.g., Fig. 6 – element of 120 – "Check Out" , "CheckOut Recursive");
- determining whether any dependent objects of said object being checked out exist based on said information;
- automatically checking-out said existing dependent objects when said object is checked-out; and providing a status to said user (e.g., Col. 11, Lines 24-29 – the VCAT system preferably determines during each check-out operation which other versionable items need to be checked out in order to modify the configuration of an item; the modification of these other versionable items may be referred to as "consequential changes.", 45-48 – because the configuration of the process is set forth in a hierarchal manner, the VCAT system must allow for checking out items having subordinate items that are also versionable).

Further, Hammack discloses the data stored in an XML document is accessed in accordance with an object model that provides for parsing the document to create a data tree structure having a plurality of nodes associated with the version control data (e.g., Col. 22, Lines 52-58), but does not explicitly disclose wherein said object being checked out is a user defined template that is derived from a preconfigured object, and

wherein said existing dependent objects are children user defined templates of said object being checked out or instances of said object or of said children user defined templates.

However, in an analogous art of *Object-Orient Versioning in a Concurrent Engineering Design Environment*, Al-Khudair discloses wherein said object being checked out is a user defined template that is derived from a preconfigured object, and wherein said existing dependent objects are children user defined templates of said object being checked out or instances of said object or of said children user defined templates (e.g., P. 107, Lines 2-3 – The checkin and checkout operations are used to deposit and retrieve design objects to/from workspaces; Fig. 7 – An object-oriented model of configuration versions; Sec. 4 – An Object-Oriented Model for Configuration Versioning, 4.1 The Model - … an object-oriented model which supports *the semantic extension of versions to cope with configuration versions* as shown in Fig. 7. This model is independent of the content of the design object and can represent a totally versioned configuration (i.e. both the configuration and its components are versioned). This enables the version model to be sufficiently generic so that it can be applied to a variety of engineering domains …; P. 113, 1st full-Para – *Configuration versioning* is the creation of a new version of the configuration … the configuration state before the change is retained as well as the new state (i.e., derived from), allowing multiple versions of a configuration to co-exist. The evolution of a configuration is represented as a hierarchy called a *configuration-derivation graph* (CDG). A *generic configuration* (i.e., preconfigured object) represents an abstract configuration which is associated with

the versioned configuration … Configuration versioning allows the instance of each class (i.e., derived from) in the standard CCH (Composite Class Hierarchy) to have multiple versions (i.e., dependent objects) …)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Al-Khudair into the Hammack's system to further provide wherein said object being checked out is a user defined template that is derived from a preconfigured object, and wherein said existing dependent objects are children user defined templates of said object being checked out or instances of said object or of said children user defined templates in the Hammack system.

The motivation is that it would further enhance the Hammack's system by taking, advancing and/or incorporating Al-Khudair's system which offers significant advantages to determine if a dynamically bound configuration version is consistent with its design goals and its assembly has no design conflicts between its components' versions as once suggested by Al-Khudair (e.g., Abstract)

9. **As to claim 6** (Original) (incorporating the rejection in claim 5), Hammack discloses the method further comprising: sorting said existing dependent objects so that parents precede children (e.g., Col. 11, Lines 48-51 – in one embodiment, if a recursive check-out or check-in is selected by the user, the VCAT system generates a dialog window that provides the user with a list of versionable, subordinate items that may be checked out (or checked-in))

10. **As to claim 7** (Currently Amended) (incorporating the rejection in claim 5), Hammack discloses the method wherein one of said existing dependent objects is a derivation child of said object (e.g., Col. 11, Lines 45-55 – because the configuration of the process is set forth in a hierarchal manner, the VCAT system must allow for checking out items having subordinate items that are also versionable)

11. **As to claim 8** (Previously Presented) (incorporating the rejection in claim 7), Hammack discloses the method further comprising: automatically checking-out a derivation child only if said derivation child is checked-in (e.g., Col. 11, Lines 45-55 – because the configuration of the process is set forth in a hierarchal manner, the VCAT system must allow for checking out items having subordinate items that are also versionable)

12. **As to claim 17** (Currently Amended), Hammack discloses a computer readable medium having executable instructions stored thereon to perform a method of automatic check-out for a source control system in a process control system, said method comprising:

- storing information associated with an object (e.g., Fig. 3; Col. 6, Lines 25-43 – with reference now to Fig. 3, the data stored in the configuration database may be presented to a user via a configuration database administrative interface such as Delta V® Explorer, which will hereinafter be referred to as “the Explorer

system"; the Explorer system sets forth a configuration hierarchy in a windows-type environment having a suite of configuration tools for modifying the elements of the hierarchy);

- receiving a request from a user to check-out said object (e.g., Fig. 6 – element of 120 – "Check Out" , "CheckOut Recursive");
- determining whether any dependent objects of said object being checked-out exist based on said information;
- automatically checking-out said existing dependent objects when said object being checked-out is checked-out; and providing a status to said user (e.g., Col. 11, Lines 24-29 – the VCAT system preferably determines during each check-out operation which other versionable items need to be checked out in order to modify the configuration of an item; the modification of these other versionable items may be referred to as "consequential changes.", 45-48 – because the configuration of the process is set forth in a hierarchal manner, the VCAT system must allow for checking out items having subordinate items that are also versionable).

Further, Hammack discloses the data stored in an XML document is accessed in accordance with an object model that provides for parsing the document to create a data tree structure having a plurality of nodes associated with the version control data (e.g., Col. 22, Lines 52-58), but does not explicitly disclose wherein said object being checked-out is a user defined template, and wherein said existing dependent objects

are children user defined templates of said object being checked-out or instances of said object being checked-out or of said children user defined templates.

However, in an analogous art of *Object-Orient Versioning in a Concurrent Engineering Design Environment*, Al-Khudair discloses wherein said object being checked-out is a user defined template, and wherein said existing dependent objects are children user defined templates of said object being checked-out or instances of said object being checked-out or of said children user defined templates (e.g., P. 107, Lines 2-3 – The checkin and checkout operations are used to deposit and retrieve design objects to/from workspaces; Fig. 7 – An object-oriented model of configuration versions; Sec. 4 – An Object-Oriented Model for Configuration Versioning, 4.1 The Model - … an object-oriented model which supports *the semantic extension of versions to cope with configuration versions* as shown in Fig. 7. This model is independent of the content of the design object and can represent a totally versioned configuration (i.e. both the configuration and its components are versioned). This enables the version model to be sufficiently generic so that it can be applied to a variety of engineering domains …; P. 113, 1st full-Para – *Configuration versioning* is the creation of a new version of the configuration … the configuration state before the change is retained as well as the new state (i.e., derived from), allowing multiple versions of a configuration to co-exist. The evolution of a configuration is represented as a hierarchy called a *configuration-derivation graph* (CDG). A *generic configuration* (i.e., preconfigured object) represents an abstract configuration which is associated with the versioned configuration … Configuration versioning allows the instance of each class (i.e., derived

from) in the standard CCH (Composite Class Hierarchy) to have multiple versions (i.e., dependent objects) …)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Al-Khudair into the Hammack's system to further provide wherein said object being checked-out is a user defined template, and wherein said existing dependent objects are children user defined templates of said object being checked-out or instances of said object being checked-out or of said children user defined templates in the Hammack system.

The motivation is that it would further enhance the Hammack's system by taking, advancing and/or incorporating Al-Khudair's system which offers significant advantages to determine if a dynamically bound configuration version is consistent with its design goals and its assembly has no design conflicts between its components' versions as once suggested by Al-Khudair (e.g., Abstract)

Claim Rejections – 35 USC § 102(b)

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102(b) that form the basis for the rejections under this section made in this office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

13. Claim 16 is rejected under 35 U.S.C. 102(b) as being anticipated by *Al-Khudair*

14. **As to claim 16** (Currently Amended), *Al-Khudair* discloses a computer readable medium having executable instructions stored thereon to perform a method version control (e.g., Abstract, Lines 1-3 – ... with tracking the evolution of design component versions and their related design configuration versions in a current engineering design environment), said method comprising:

- When checking-in an object, determining relationships of said object by (e.g., P. 107, 1st Para - … The checkin ... used to deposit ... design object to ... workspaces; 3rd full Para - Consistency constraints are automatically checked in multi-version configuration at two levels … a component version … a configuration version … only when constraint checking at the configuration level reveals a consistent configuration is the consistency state of the component version set to consistent …):
- determining whether said object being checked-in has a first derivation parent (e.g., P. 110, 1st full Para - … to allow **version merging** (i.e., more than one parent of a version). Hence, a versioned object consists of a hierarchy of versions call a version-derivation graph (VDG) connected **by the Derived-From** (or **Parent/Child**) **relationships** …);
- adding a name and a version of said first derivation parent to a list of object relationships, if said object being checked-in has said first derivation parent (e.g., P. 118, 1st Para - … When deriving a new version of a configuration, the component name along with the specific version number should be specified ...);

- determining for each contained object that is contained in said object being checked-in, whether said contained object has a second derivation parent, if said object being checked-in does not have said first derivation parent (e.g., P. 118, 1st Para - …It is also possible to have **an object version derived from two or more existing version of an object** (version merging) ...);
- adding a name and a version of said second derivation parent to said list of object relationships, if said contained object being checked-in has said second derivation parent; and providing said list of object relationships (e.g., P. 118, 1st Para - … When deriving a new version of a configuration, the **component name** along with the **specific version number** should be specified ...; Sec. 4.1.2 Model Basic Operations, 2nd Para – The operations specific to versioned objects/configurations are:

  - Simple object

    DERIVE <object name>.<version number>

    FROM {list of parents})

  - Configuration

    DERIVE <configuration name>.<version number>

    FROM {list of parents (component name>.<version number>}

)

Conclusion

15. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ben C. Wang whose telephone number is 571-270-1240. The examiner can normally be reached on Monday - Friday, 8:00 a.m. - 5:00 p.m., EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on 571-272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Ben C Wang/
Examiner, Art Unit 2192
September 26, 2008

/Tuan Q. Dam/
Supervisory Patent Examiner, Art Unit 2192